Claims

The following is a copy of Applicant's claims that identifies language being added with underlining ("___") and language being deleted with strikethrough ("~~—~~"), as is applicable:

1-7. (Canceled)

8. (Currently amended) The method of claim ~~1~~17, further comprising ~~the step of~~ dynamically receiving information about unavailable hardware functionality and replacement instructions that are configured to replace original program instructions that require the unavailable hardware functionality.

9-12. (Canceled)

13. (Currently amended) A dynamic patching program stored on a computer-readable medium, the program comprising:

logic configured to gain control over execution of a program;

logic configured to intercept original program instructions during program execution;

logic configured to determine if associated instructions have been cached in a code cache controlled by the dynamic patching program;

logic configured to execute associated instructions that have been cached in the code cache;

logic configured to determine if ~~a~~ an original program instruction requires unavailable hardware functionality; and

logic configured to dynamically replace ~~the~~ original program instructions with ~~a~~ replacement instructions that ~~does~~ do not require unavailable hardware functionality if it is determined that the program instructions ~~requires~~ require unavailable hardware functionality, wherein the logic configured to dynamically replace is configured to fetch a replacement instruction, store the replacement instruction in the code cache, and execute the replacement instruction from the code cache, wherein the logic configured to dynamically replace does not replace or translate original program instructions that do not require unavailable hardware functionality.

14-15. (Canceled)

16. (Original) The system of claim 13, further comprising logic configured to dynamically receive information about unavailable hardware functionality and replacement instructions that are configured to replace original program instructions that require the unavailable hardware functionality.

17. (Currently amended) A method for dynamically patching code, comprising ~~the steps of~~:

gaining control over the execution of a program using a software interface;

intercepting original program instructions during execution of the program using the software interface;

determining whether ~~the program~~ associated instructions have been cached in a code cache of the software interface and, if so, executing the cached instructions from the code cache;

if ~~the program~~ associated instructions have not been cached, determining if the original program instructions require unavailable hardware functionality; and

dynamically replacing the original program instructions with replacement instructions that do not require unavailable hardware functionality if it is determined that the original program instructions require unavailable hardware functionality, the dynamic replacing comprising fetching replacing instructions, storing the replacement instructions in the code cache, and executing the replacement instructions from the code cache;

wherein original program instructions that do not require unavailable hardware functionality are not replaced with replacement instructions or translated.

18. (Canceled)

19. (Currently amended) The method of claim 18, wherein ~~the step of~~ dynamically replacing the original program instructions further comprises changing all references to replaced original program instructions such that the ~~executing the~~ replacement instructions in the code cache are executed in lieu of the replaced original program instructions each time ~~a functionality associated with~~ the replaced original program instructions ~~is required~~ would have been executed, thereby avoiding the need to again fetch replacement instructions for the original program instructions.

20. (Currently amended) The method of claim 19, wherein the replacement instructions comprise part of a patch that is made available to the software interface via an application programming interface.

21. (Currently amended) A dynamic execution layer interface (DELI) residing between an application and computing system hardware, comprising:

a transparent mode layer that is configured to gain control over the operation of the application and to fetch replacement instructions that are to replace existing application instructions;

a system control and configuration layer configured to provide policies for the replacement of existing application instructions with the replacement instructions;

a core configured to, during execution of the application, receive policies for the replacement of existing application instructions, to determine whether the existing application instructions require unavailable hardware functionality, to receive replacement instructions to be executed in lieu of the existing application instructions that do require unavailable hardware functionality, to ~~dynamically~~ cache the replacement instructions, and to execute the replacement instructions, wherein existing application instructions that do not require unavailable hardware functionality are not translated or replaced; and

a code cache in which the replacement instructions are cached and from which the replacement instructions are executed.

22. (Original) The DELI of claim 21, wherein the transparent mode layer is further configured to fetch application instructions from the application and wherein the core is further configured to cache fetched application instructions in the code cache.

23. (New) The system of claim 13, further comprising logic configured to change all references to replaced original program instructions with references to replacement instructions such that replacement instructions in the code cache are executed in lieu of the replaced original program instructions without the need to again fetch the replacement instructions.

24. (New) The DELI of claim 21, wherein the core is further configured to change all references to replaced original program instructions with references to replacement instructions such that replacement instructions in the code cache are executed in lieu of the replaced original program instructions without the need to again fetch the replacement instructions.

25. (New) The method of claim 17, further comprising copying intercepted original program instructions that do not require unavailable hardware functionality to the software interface code cache such that substantially all execution ultimately occurs within the code cache.

26. (New) The program of claim 13, further comprising logic configured to copy intercepted original program instructions that do not require unavailable

hardware functionality to the code cache such that substantially all execution ultimately occurs within the code cache.

27. (New) The DELI of claim 21, wherein the core is further configured to copy intercepted original program instructions that do not require unavailable hardware functionality to the DELI code cache such that substantially all execution ultimately occurs within the code cache.